This listing of claims will replace all prior versions, and listings, of claims in the application.

Listing of Claims:

1.   (Currently amended) A <u>computer-implemented</u> method of processing runtime functions, comprising:

compiling code to produce executable code that is marked with an identifier indicating that the executable code <u>comprises an object file containing a list of valid target addresses for use in implementing</u> ~~supports~~ runtime protection;

<u>storing in a table, the list of valid target addresses as a reference list of valid target addresses;</u>

receiving a call to a runtime function of the executable code;

determining associated data from the call to the runtime function;

determining a target address from the associated data;

comparing the target address with ~~a~~ <u>the</u> reference list of valid target addresses <u>stored in the table</u>;

if the target address is found on the reference list of valid target addresses then executing the runtime function; and

if the target address is not found on the reference list of valid target addresses then terminating execution of the ~~executable code~~ <u>runtime function</u>.

2.   (Original) The method of claim 1, wherein the step of determining the associated data comprises accessing data in a data structure connected with the runtime function and calculating the associated data based on the accessed data.

3-5   (Canceled).

6.   (Previously presented) The method of claim 1 comprising the step of generating the reference list of valid target addresses during execution of a previous runtime function.

7-9.   (Canceled)

10.   (Currently amended) A computer-readable <u>storage</u> medium having stored thereon computer-executable instructions for performing a method of processing runtime functions, the method comprising:

      receiving a call to a runtime function;

      determining associated data from the call to the runtime function;

      ~~determining a target address from the associated data~~ <u>deriving a security cookie by XORing a secret value with each of the values retrieved from a jmp_buf buffer, the retrieved values precluding a first security cookie that has been stored previously in the jmp_buf buffer</u>;

      comparing the ~~target address with a reference list of valid target addresses~~ <u>derived security cookie against the first security cookie</u>; and

      if the ~~target address is found on the reference list of valid target addresses~~ <u>derived security cookie matches the first security cookie</u> then executing the runtime function; and

      if the ~~target address is not found on the reference list of valid target addresses~~ <u>derived security cookie does not match the first security cookie</u> then terminating execution of the runtime function.

11.   (Currently amended) The computer-readable <u>storage</u> medium of claim 10, wherein the step of determining the associated data comprises accessing data in a data structure connected with the runtime function and calculating the associated data based on the accessed data.

12-14. (Canceled)

15.   (Currently amended) The computer-readable <u>storage</u> medium of claim 10 ~~comprising the step of generating the list of target addresses~~ <u>wherein the first security cookie is derived</u> during execution of a previous runtime function.

16-18. (Canceled)

19.   (Currently amended) A system for processing runtime functions, comprising:

          a compiler that compiles code to produce an executable that is marked with an identifier

indicating that the executable comprises an object file containing a list of valid target addresses

for use in implementing supports runtime protection;

          a processor that receives a call to a runtime function; and

          a dispatcher system that determines associated data from the call to the runtime function,

determines a target address from the associated data, and if the target address is found on the

reference list of valid target addresses then executes the target runtime function.

20.   (Original) The system of claim 19, wherein the dispatcher system comprises a module to

access data in a data structure connected with the runtime function and calculate the associated

data based on the accessed data.

21-22. (Canceled)

23.   (Currently amended) The system of claim 19, further comprising a compiler that

generates the reference list of valid target addresses.

24-27. (Canceled)

28.   (Currently amended) The method of claim 1 comprising wherein the step of storing in the

table comprises storing the target address in a caller provided location during execution of a

previous runtime function.

29-36. (Canceled)

37. (Currently amended)   The method of claim 1, further comprising:

     determining if at least a portion of the associated data is valid; and

     preventing execution of the ~~target~~ <u>runtime function</u> if the associated data is not valid.

38. (Previously presented)   The method of claim ~~3~~ <u>37</u>, wherein the step of determining if the associated data is valid comprises retrieving a security cookie from the associated data and comparing the retrieved security cookie to a list of valid security cookies.

39. (Currently amended)   The method of claim ~~3~~ <u>37</u>, further comprising determining and storing a predetermined calculated value based on at least a portion of the associated data, prior to receiving the call to the runtime function.

40. (Currently amended)   The method of claim ~~8~~ <u>39</u>, wherein determining if the associated data is valid comprises comparing the predetermined calculated value to another calculated value based on the associated data.

41. (Currently amended) The computer readable medium of claim 10, having further computer-executable instructions for determining if at least a portion of the associated data is valid, and preventing execution of the ~~target~~ <u>runtime function</u> if the associated data is not valid.

42. (Canceled)

43. (Currently amended) The computer-readable medium of claim ~~12~~ <u>41</u>, having further computer-executable instructions for determining and storing a predetermined calculated value based on at least a portion of the associated data, prior to receiving the call to the runtime function.

44. (Currently amended) The computer-readable medium of claim ~~17~~ <u>43</u>, wherein determining

if the associated data is valid comprises comparing the predetermined calculated value to another
calculated value based on the associated data.

45. (Currently amended) The system of claim 19, wherein the dispatcher system comprises
modules to determine if at least a portion of the associated data is valid and prevent execution of
the ~~target~~ runtime function if the associated data is not valid.

46. (Currently amended) The system of claim ~~21~~ 45, further comprising a storage device that
stores ~~a~~ the list of valid ~~targets~~ target addresses~~, wherein the dispatcher system determines if the~~
~~associated data is valid by comparing the target address to the list of valid target addresses~~.

47. (Currently amended) The system of claim ~~21~~ 45, wherein the dispatcher system determines
if the associated data is valid by retrieving a security cookie from the associated data and
comparing the retrieved security cookie to a list of valid security cookies.

48. (Currently amended) The system of claim ~~21~~ 45, wherein the processor determines and
stores a predetermined calculated value based on at least a portion of the associated data, prior to
receiving the call to the runtime function.

49. (Currently amended) The system of claim ~~25~~ 48, wherein the dispatcher system determines
if the associated data is valid by comparing the predetermined calculated value to another
calculated value based on the associated data.

50. (New) The method of claim 1, wherein the identifier is an identifier bit that is operable to be
set for indicating that the executable code comprises the object file containing the list of valid
target addresses for use in implementing runtime protection.

51. (New) The method of claim 1, wherein the table is a .setjmp table and the call to the runtime